15-281 Artificial Intelligence HW4

INSTRUCTIONS

- **Due: Tuesday, February 11, 2020 at 10:00 PM EDT.** Remember that you may use up to 2 slip days for the Written Homework making the last day to submit **Thursday, February 13, 2020 at 10:00 PM EDT**.
- **Format:** Submit the answer sheet pdf containing your answers. You should solve the questions on this handout (either through a pdf annotator, or by printing, then scanning). Make sure that your answers (typed or handwritten) are within the dedicated regions for each question/part. If you do not follow this format, we may deduct points.
- **How to submit:** Submit a pdf with your answers on Gradescope. Log in and click on our class 15-281 and click on the submission titled HW4 and upload your pdf containing your answers.
- **Policy:** See the course website for homework policies and Academic Integrity.

Name	
Andrew ID	
Hours to complete?	

For staff use only

Q1	Q2	Q3	Q4	Q5	Q6	Total
/16	/16	/8	/20	/16	/24	/100

Q1. [16 pts] Local Search - Warm Up

Mark *True* or *False* for each of the statements below.

(a) [1 pt] Regular hill climbing is optimal.

◯ True ◯ False

(b) [1 pt] Random restart hill climbing is optimal.

◯ True ◯ False

(c) [1 pt] Simulated annealing allows for downward moves according to some fixed constant temperature T.

◯ True ◯ False

(d) [1 pt] Simulated annealing is generally less efficient than random walk.

◯ True ◯ False

(e) [1 pt] In genetic algorithms, each two parent states go through crossover to create one offspring.

◯ True ◯ False

(f) [1 pt] Local search algorithms always maintain the same number of states in every iteration.

◯ True ◯ False

For each question below, select the name of the algorithm that most closely resembles the special case.

(g) [2 pts] Local beam search with k = 1.

◯ A) hill-climbing search
◯ B) random-walk search
◯ C) breadth-first search

(h) [2 pts] Local beam search with one initial state and no limit on the number of states retained.

◯ A) hill-climbing search
◯ B) random-walk search
◯ C) breadth-first search

(i) [2 pts] Simulated annealing with $T = 0$ at all times (and omitting the termination test).

◯ A) hill-climbing search
◯ B) random-walk search
◯ C) breadth-first search

(j) [2 pts] Simulated annealing with $T = \infty$ at all times.

◯ A) hill-climbing search
◯ B) random-walk search
◯ C) breadth-first search

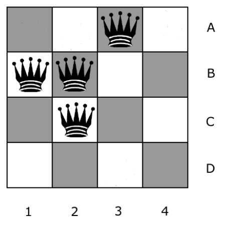
(k) [2 pts] Genetic algorithm with population size $N = 1$.

◯ A) hill-climbing search
◯ B) random-walk search
◯ C) breadth-first search

Q2. [16 pts] 4-Queens Problem

The following images show pseudocode for a hill-climbing algorithm to the 4-Queens problem and a starting 4-Queens problem state.

```
function 4-QUEENS-HILL-CLIMBING(problem)
    current_state = problem.INITIAL-STATE
    loop do
        neighbor = state with lowest number of conflicts between queens obtained by
                   making one legal chess move from current_state
        if neighbor.num_conflicts >= current_state.num_conflicts
            return current_state
        current_state = neighbor
```



(a) [4 pts] Write down the states of each of the 4 queens after one iteration of the loop.

States:

(b) [6 pts] Will the hill climbing algorithm presented above stop for the given starting state? If so, what does the state end up as? Please check all that apply.

☐ A) Will not stop

☐ B) Local optima

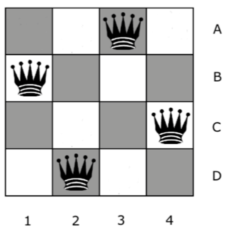
☐ C) Global optima

☐ D) Shoulder

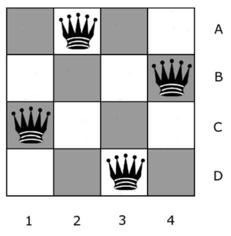
☐ E) None of the above

(c) [6 pts] We can use a genetic algorithm to solve the 4-Queens problem. In our algorithm, a crossover state is a state combining the top half of one board and the bottom half of another. A state can be crossed over with any other state in the population. A mutation occurs when one queen is moved horizontally for any number of spaces. The following image shows the two solutions (goal states) to the 4-Queens problem.

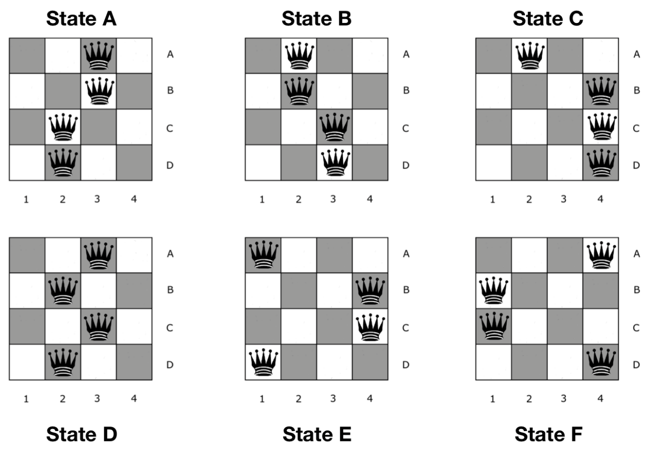
Goal 1



Goal 2



We want to reach one of the goal states above using the genetic algorithm. Our current population of states is shown in the following image above. With which of these states in the current population can we reach a goal state through 1 crossover and 1 mutation?



☐ (A, B)

☐ (B, C)

☐ (C, D)

☐ (C, F)

☐ (E, F)

☐ None of the above

Q3. [8 pts] Malicious PDFs

Cybersecurity attacks often occur through PDFs containing malware, and an unsuspecting user can easily be exploited upon opening the malicious file. Thankfully, people have developed PDF malware detectors that help us detect such threats. But, the detectors are not perfect and can be hacked. You have noticed that there is a popular detector being used by a lot of users. However, you are concerned about the robustness of the detector. You decide to stand in the shoes of a hacker to design a malicious PDF that can evade the detector.

If you succeed in doing so, you can remind the developers and the users that they need to develop a better detector. After listening to class lecture, you decide to hack the detector using a genetic algorithm. The initial population is a set of malicious PDF samples. The mutation occurs with a probability (≤ 0.5) and is either a deletion, insertion, or replacement of specific PDF object (e.g. header).

Assume that you have access to a function ***isMalicious(d)***, which returns ***1*** if the input PDF ***d*** is malicious, and ***0*** otherwise. You also have a separate malware detector ***c(d)*** that returns its predicted likelihood that ***d*** is malicious by returning values in the range ***[0,1]*** (i.e. a value of ***1*** indicates full confidence in a malicious PDF). So, ***isMalicious(d)*** returns the ground truth (which can be implemented by running expensive tests), and the malware detector ***c(d)*** returns a prediction.

Your task is to find a PDF through a simple genetic algorithm (with mutation operation only, and no cross-over), such that it retains its malicious behavior but evades the malware detector. We say a malicious PDF successfully evades the detector if ***c(d) < 0.5*** .

Recall that the fitness function should return higher values for better states, which will be selected for continued evolution. **Select all of the following fitness function(s)** that would be reasonable in this context.

(*Hint:* Our fitness function should assign higher values to PDFs that achieve the goal of retaining their maliciousness while simultaneously evading the malware detector.)

☐ A)
$$f(d) = \begin{cases} 0.5 - c(d) & isMalicious(d)=1 \\ -10 & isMalicious(d)=0 \end{cases}$$

☐ B)
$$f(d) = \begin{cases} c(d) & isMalicious(d)=1 \\ -100 & isMalicious(d)=0 \end{cases}$$

☐ C)
$$f(d) = \begin{cases} (-1)\times c(d) & isMalicious(d)=1 \\ -100 & isMalicious(d)=0 \end{cases}$$

☐ D)
$$f(d) = c(d)$$

☐ E)
$$f(d) = -c(d)$$

☐ F)

None of the above

Q4. [20 pts] Two Mines Company

The Two Mines Company own two different mines that produce an ore which, after being crushed, is graded into three classes: high-, medium- and low-grade. The company has contracted to provide a smelting plant with ***at least*** 12 tons of high-grade, ***at least*** 8 tons of medium-grade and ***at least*** 24 tons of low-grade ore per week. The two mines have different operating characteristics as detailed below. Additionally, we cannot work the same mine for more than 6 days a week.

How many days per week should each mine be operated to fulfill the smelting plant contract?

Note: We have (implicitly) assumed that it is permissible to work in fractions of days. We also assume that both mines can be operated on the same day.

Mine name	**Cost per day**	**High-grade per day**	**Medium-grade per day**	**Low-grade per day**
Heigh Ho	180	4 tons	5 tons	4 tons
Kessel	160	2 tons	4 tons	8 tons

(a) [8 pts] Write this problem as an LP in **inequality form** as defined in lecture. Define any variables you use in your formulation. *Warning:* Be sure to strictly follow the inequality form, including the proper use of less than or equal, or you will lose points.

Inequality Form:

(b) [8 pts] Accurately plot the graphical representation of this linear program. Specifically:

- Plot the boundary of each halfspace as a line (no need to shade or draw normal vectors), and
- Plot the cost vector as an arrow with *magnitude one*, somewhere within the feasible region.

Do *not* draw; use a plotting tool such as Python matplotlib and include the resulting image here. Be sure to label the axes of your plot, including tick-marks. Display your plot with a *square* aspect ratio, e.g. in matplotlib: `plt.axis("equal")`. Additionally, zoom your plot to make the entire feasible region visible.

Tip to a plot vector $[v_1, v_2]^T$ in matplotlib starting at some point (x_1, x_2):

```
plt.quiver(x1, x2, v1, v2, angles="xy", scale_units="xy", scale=1)
```

Tip to properly control scaling using a specific width and height:

```
plt.figure(figsize=(width,height))
```

Plot:

(c) [4 pts] Find the optimal solution to the LP problem. Give the solution as days per week per mine as well as the corresponding cost.

Heigh Ho:	**Kessel:**	**Cost:**

Q5. [16 pts] Graphing LPs

For the inequality form of a linear program, and a given A matrix and $\boldsymbol{b}$ vector,

$$\begin{aligned} \min_{\boldsymbol{x}}.\ & \boldsymbol{c}^T\boldsymbol{x} \\ \text{s.t. } & A\boldsymbol{x} \preceq \boldsymbol{b} \end{aligned}$$

For each row i of A and $\boldsymbol{b}$, accurately plot 1) the line $a_{i,1}x_1 + a_{i,2}x_2 = b_i$ and 2) the vector $[a_{i,1}, a_{i,2}]^T$ as an arrow beginning at any point on its respective line.

Tip to plot a vector $[v_1, v_2]^T$ in matplotlib starting at some point (x_1, x_2):

```
plt.quiver(x1, x2, v1, v2, angles="xy", scale_units="xy", scale=1)
```

Tip to properly control scaling using a specific width and height:

```
plt.figure(figsize=(width,height))
```

Do *not* draw; use a plotting tool such as Python matplotlib and include the resulting image here. Be sure to label the axes of your plot, including tick-marks. Display your plot with a *square* aspect ratio, e.g. in matplotlib: `plt.axis("equal")`. Additionally, zoom your plot or adjust the axes such that all of the vectors are visible. You do not need to shade the feasible regions.

(a) [8 pts]

$$A = \begin{bmatrix} 3 & 5 \\ 7 & 6 \\ 12 & 6 \end{bmatrix} \qquad \boldsymbol{b} = \begin{bmatrix} 10 \\ 17 \\ 27 \end{bmatrix}$$

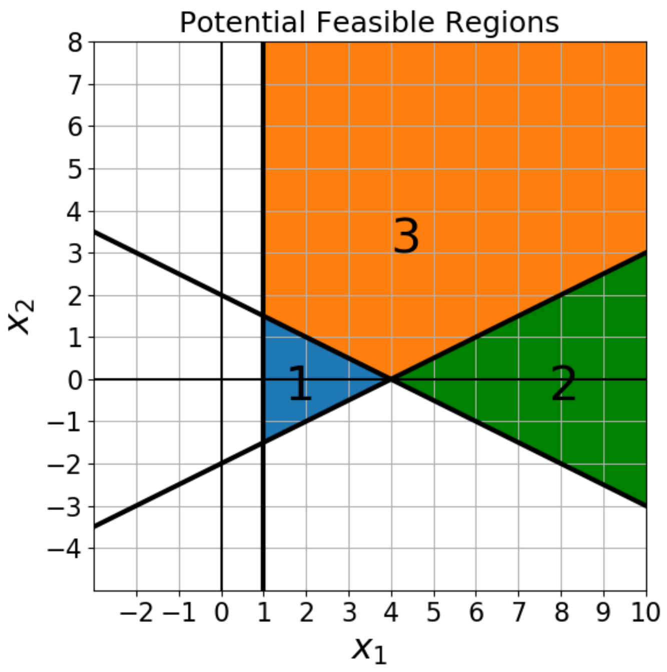
Plot:

(b) [8 pts]

$$A = \begin{bmatrix} -2 & -1 \\ 2 & 5 \\ 7 & 2 \end{bmatrix} \qquad \boldsymbol{b} = \begin{bmatrix} -4 \\ 10 \\ 11 \end{bmatrix}$$

Plot:

Q6. [24 pts] Feasible Regions



In this problem, you are given a graph with constraint boundary lines (**bolded**) and potential feasible regions. You may assume shaded regions at the edge of the plot continue to infinity. Provide the corresponding A and $\boldsymbol{b}$ based on the inequality form below for each feasible region in the boxes below:

$$\begin{aligned} \min_{\boldsymbol{x}}.\ & \boldsymbol{c}^T \boldsymbol{x} \\ \text{s.t.}\ & A\boldsymbol{x} \preceq \boldsymbol{b} \end{aligned}$$

(a) [8 pts] Feasible Region 1

A**:**	$\boldsymbol{b}$**:**

(b) [8 pts] Feasible Region 2

A**:**	$\boldsymbol{b}$**:**

(c) [8 pts] Feasible Region 3

A**:**	$\boldsymbol{b}$**:**